rastools Documentation

Release 0.5

Dave Hughes

April 15, 2016

Contents

rastools is a small suite of utilities for converting data files obtained from SSRL (Stanford Synchrotron Radiation Lightsource) scans (.RAS and .DAT files) into images. Various simple manipulations (cropping, percentiles, histograms, color-maps) are supported. Most tools are command line based, but a Qt-based GUI is also included.

CHAPTER 1

Contents

1.1 Installation

rastools is distributed in several formats. The following sections detail installation on a variety of platforms.

1.1.1 Pre-requisites

Where possible, I endeavour to provide installation methods that provide all pre-requisites automatically - see the following sections for platform specific instructions.

If your platform is not listed (or you're simply interested in what rastools depends on): rastools depends primarily on matplotlib. If you wish to use the GUI you will also need PyQt4 installed.

Additional optional dependencies are:

- xlwt - required for Excel writing support
- GIMP - required for GIMP (.xcf) writing support

1.1.2 Ubuntu Linux

For Ubuntu Linux, it is simplest to install from the PPA as follows (this also ensures you are kept up to date as new releases are made):

```
$ sudo add-apt-repository ppa://waveform/ppa
$ sudo apt-get update
$ sudo apt-get install rastools
```

1.1.3 Microsoft Windows

On Windows it is simplest to install from the standalone MSI installation package available from the homepage. Be aware that the installation package requires administrator privileges.

1.1.4 Apple Mac OS X

XXX To be written

1.1.5 Other Platforms

If your platform is *not* covered by one of the sections above, rastools is available from PyPI and can therefore be installed with the Python distribute `pip` tool:

```
$ pip install rastools
```

Theoretically this should install the mandatory pre-requisites, but optional pre-requisites require suffixes like the following:

```
$ pip install "rastools[GUI,XLS]"
```

Please be aware that at this time, the PyQt package does not build "nicely" under `pip`. If it is available from your distro's package manager I strongly recommend using that as your source of this pre-requisite.

If PyQt is not provided by your distro (or you're on some esoteric platform without a package manager), you can try following the instructions on the Veusz wiki for building PyQt (and SIP) under a virtualenv sandbox.

1.1.6 Development

If you wish to develop rastools, you can install the pre-requisites, construct a virtualenv sandbox, and check out the source code from GitHub with the following command lines:

```
# Install the pre-requisites
$ sudo apt-get install python-matplotlib python-xlwt python-qt4 python-virtualenv python-sphinx g

# Construct and activate a sandbox with access to the packages we just
# installed
$ virtualenv --system-site-packages sandbox
$ source sandbox/bin/activate

# Check out the source code and install it in the sandbox for development and testing
$ git clone https://github.com/waveform80/rastools.git
$ cd rastools
$ make develop
```

The above instructions assume you are on Ubuntu Linux. Please feel free to extend this section with instructions for alternate platforms.

1.2 rasdump

This utility accepts a QSCAN RAS file and an optional channel definition file. For each channel listed in the latter, a dump is produced of the corresponding channel in the RAS file. Various options are provided for customizing the output including percentile limiting, and output format.

1.2.1 Synopsis

```
$ rasdump [options] data-file [channels-file]
```

1.2.2 Description

Dump channel data from *data-file*. The optional *channel-file* defines the indices and names of the channels to dump. If the *channel-file* is omitted all channels are extracted and channels in RAS files will be unnamed.

--version
show program's version number and exit

-h, --help
show this help message and exit

-q, --quiet
produce less console output

-v, --verbose
produce more console output

-l LOGFILE, **--log-file**=LOGFILE
log messages to the specified file

-P, --pdb
run under PDB (debug mode)

--help-formats
list the available file output formats

-p PERCENTILE, **--percentile**=PERCENTILE
clip values in the output image to the specified low-high percentile range (mutually exclusive with *-r*)

-r RANGE, **--range**=RANGE
clip values in the output image to the specified low-high count range (mutually exclusive with *-p*)

-C CROP, **--crop**=CROP
crop the input data by left,top,right,bottom points

-e, --empty
if specified, include empty channels in the output (by default empty channels are ignored)

-o OUTPUT, **--output**=OUTPUT
specify the template used to generate the output filenames; supports {variables}, see –help-formats for supported file formats. Default: {filename_root}_{channel:02d}_{channel_name}.csv

-m, --multi
if specified, produce a single output file with multiple pages or sheets, one per channel (only available with certain formats)

1.2.3 Examples

Basic Usage

The most basic usage of rasdump is to specify only the RAS file from which to dump data. This will dump data in the default CSV format, one file per channel with no cropping and no percentile limiting. All channels (except empty ones) will be extracted, and will be anonymous (since no channels file has been specified to name them):

```
$ rasdump JAN12_CHINAFISH_LZ_003.RAS
Writing channel 0 () to JAN12_CHINAFISH_LZ_00_.csv
Channel 0 () is empty, skipping
Writing channel 1 () to JAN12_CHINAFISH_LZ_01_.csv
Writing channel 2 () to JAN12_CHINAFISH_LZ_02_.csv
Writing channel 3 () to JAN12_CHINAFISH_LZ_03_.csv
Writing channel 4 () to JAN12_CHINAFISH_LZ_04_.csv
Writing channel 5 () to JAN12_CHINAFISH_LZ_05_.csv
Writing channel 6 () to JAN12_CHINAFISH_LZ_06_.csv
Writing channel 7 () to JAN12_CHINAFISH_LZ_07_.csv
Writing channel 8 () to JAN12_CHINAFISH_LZ_08_.csv
Writing channel 9 () to JAN12_CHINAFISH_LZ_09_.csv
Writing channel 10 () to JAN12_CHINAFISH_LZ_10_.csv
Writing channel 11 () to JAN12_CHINAFISH_LZ_11_.csv
Writing channel 12 () to JAN12_CHINAFISH_LZ_12_.csv
Writing channel 13 () to JAN12_CHINAFISH_LZ_13_.csv
Writing channel 14 () to JAN12_CHINAFISH_LZ_14_.csv
Writing channel 15 () to JAN12_CHINAFISH_LZ_15_.csv
```

Help Lists

XXX To be written

Substitution Templates

XXX To be written

Advanced Usage

XXX To be written

1.3 rasextract

This utility accepts a QSCAN RAS file and an optional channel definition file. For each channel listed in the latter, an image is produced from the corresponding channel in the RAS file. Various options are provided for customizing the output including percentile limiting, color-mapping, and drawing of axes and titles.

1.3.1 Synopsis

```
$ rasextract [options] data-file [channel-file]
```

1.3.2 Description

Extract channel data from *data-file* as images. The optional *channel-file* defines the indices and names of the channels to extract. If the *channel-file* is omitted all channels are extracted and channels in .RAS files will be unnamed.

--version
show program's version number and exit

-h, --help
show this help message and exit

-q, --quiet
produce less console output

-v, --verbose
produce more console output

-l LOGFILE, **--log-file**=LOGFILE
log messages to the specified file

-P, --pdb
run under PDB (debug mode)

--help-colormaps
list the available colormaps

--help-formats
list the available file output formats

--help-interpolations
list the available interpolation algorithms

-p PERCENTILE, **--percentile**=PERCENTILE
clip values in the output image to the specified low-high percentile range (mutually exclusive with *-r*)

-r RANGE, **--range**=RANGE
clip values in the output image to the specified low-high count range (mutually exclusive with *-p*)

-C CROP, **--crop**=CROP
crop the input data by left,top,right,bottom points

-e, --empty
if specified, include empty channels in the output (by default empty channels are ignored)

-a, --axes
draw the coordinate axes in the output

-b, --color-bar
draw a color-bar showing the range of the color-map to the right of the output

-g, --grid
draw grid-lines overlayed on top of the image

-R RESIZE, **--resize**=RESIZE
resize the image; if specified as a single it is considered a multiplier for the original dimensions, otherwise two comma-separated numbers are expected which will be treated as new X,Y dimensions for the image data (note: only the image data will be resized to these dimensions, auxilliary elements like the histogram will be continue to be sized relative to the image data)

-H, --histogram
draw a histogram of the channel values below the output

--histogram-bins=BINS
specify the number of bins to use when constructing the histogram (default=32)

-c CMAP, **--colormap**=CMAP
the colormap to use in output (e.g. gray, jet, hot); see *--help-colormaps* for listing

-i INTERPOLATION, **--interpolation**=INTERPOLATION
force the use of the specified interpolation algorithm; see *--help-interpolations* for listing

-O AXES_OFFSET, **--offset**=AXES_OFFSET
specify the X,Y offset of the coordinates displayed on the axes; if one value is specified it is used for both axes

-S AXES_SCALE, **--scale**=AXES_SCALE
specify the X,Y multipliers to apply to the post-offset axes coordinates (see –offset); if one value is specified it is used for both axes

-t TITLE, **--title**=TITLE
specify the template used to display a title at the top of the output; supports `{variables}` produced by *rasinfo -t*

--x-title=TITLE_X
specify the title for the X-axis; implies –axes

--y-title=TITLE_Y
specify the title for the Y-axis; implies –axes

-o OUTPUT, **--output**=OUTPUT
specify the template used to generate the output filenames; supports `{variables}`, see *--help-formats* for supported file formats. Default: `{filename_root}_{channel:02d}_{channel_name}.png`

-m, --multi
if specified, produce a single output file with multiple layers or pages, one per channel (only available with certain formats)

1.3.3 Examples

Basic Usage

The most basic usage of rasextract is to specify only the RAS file from which to extract images. This will extract the images in the default PNG format, with the default 'gray' colormap, no cropping, no axes, no histogram, no colorbar, and no title. Furthermore all channels (except empty ones) will be extracted, and will be anonymous (since no channels file has been specified to name them):

```
$ rasextract JAN12_CHINAFISH_LZ_003.RAS
Writing channel 0 () to JAN12_CHINAFISH_LZ_00_.png
Channel 0 () is empty, skipping
Writing channel 1 () to JAN12_CHINAFISH_LZ_01_.png
Writing channel 2 () to JAN12_CHINAFISH_LZ_02_.png
Writing channel 3 () to JAN12_CHINAFISH_LZ_03_.png
Writing channel 4 () to JAN12_CHINAFISH_LZ_04_.png
Writing channel 5 () to JAN12_CHINAFISH_LZ_05_.png
Writing channel 6 () to JAN12_CHINAFISH_LZ_06_.png
Channel 6 () has no values below 30
Writing channel 7 () to JAN12_CHINAFISH_LZ_07_.png
Writing channel 8 () to JAN12_CHINAFISH_LZ_08_.png
Writing channel 9 () to JAN12_CHINAFISH_LZ_09_.png
Writing channel 10 () to JAN12_CHINAFISH_LZ_10_.png
Writing channel 11 () to JAN12_CHINAFISH_LZ_11_.png
Writing channel 12 () to JAN12_CHINAFISH_LZ_12_.png
Writing channel 13 () to JAN12_CHINAFISH_LZ_13_.png
Channel 13 () has no values below 62
Writing channel 14 () to JAN12_CHINAFISH_LZ_14_.png
Writing channel 15 () to JAN12_CHINAFISH_LZ_15_.png
Channel 15 () has no values below 1522
```

The following command line was used to extract 14 channels of data from a RAS file, crop the channels by 15 elements at the left and right, limit the data to the 95th percentile, and generate output images including axes with the standard MATLAB "jet" colormap:

```
$ rasextract -a -C 0,15,0,15 -c jet -p 95 JAN12_CHINAFISH_HZ_001.RAS channels.txt
File contains 16 channels, extracting channels 1,2,3,4,5,6,7,8,9,10,11,12,13,14
Writing channel 1 (Cu) to JAN12_CHINAFISH_HZ_01_Cu.png
Writing channel 2 (Zn) to JAN12_CHINAFISH_HZ_02_Zn.png
Writing channel 3 (Pbli) to JAN12_CHINAFISH_HZ_03_Pbli.png
Writing channel 4 (Pbla) to JAN12_CHINAFISH_HZ_04_Pbla.png
Writing channel 5 (Pblb) to JAN12_CHINAFISH_HZ_05_Pblb.png
Writing channel 6 (Ca) to JAN12_CHINAFISH_HZ_06_Ca.png
Writing channel 7 (Br) to JAN12_CHINAFISH_HZ_07_Br.png
Writing channel 8 (Mn) to JAN12_CHINAFISH_HZ_08_Mn.png
Writing channel 9 (Fe) to JAN12_CHINAFISH_HZ_09_Fe.png
Writing channel 10 (Tika) to JAN12_CHINAFISH_HZ_10_Tika.png
Writing channel 11 (Tikb) to JAN12_CHINAFISH_HZ_11_Tikb.png
Writing channel 12 (ES) to JAN12_CHINAFISH_HZ_12_ES.png
Writing channel 13 (ICR) to JAN12_CHINAFISH_HZ_13_ICR.png
Writing channel 14 (Ni) to JAN12_CHINAFISH_HZ_14_Ni.png
```

Help Lists

The various color maps available can be listed with the *--help-colormaps* option, but a more visually useful listing of the maps can be found on the matplotlib site. As can be seen above other help options also exist to, for example, list the available image formats:

```
$ rasextract --help-formats
The following file formats are available:

.bmp
.eps
.gif
.jpeg
.jpg
.pdf
.png
.ps
.svg
```

```
.svgz
.tif
.tiff
.xcf
```

Note that, depending on your installation and the availability of certain external utilities (like GIMP) certain formats may not be available.

Substitution Templates

The *-o* and *-t* options can be used to specify output filenames and titles to write into the images, respectively. Both options accept a number of "templates" which will be substituted for certain variables at runtime. The templates which are available can be discovered by running the rasinfo tool against your .RAS file (and optional channels definition) with the *rasinfo -t* option. For example:

```
$ rasinfo -t JAN12_CHINAFISH_LZ_003.RAS
{rasfile}=JAN12_CHINAFISH_LZ_003.RAS
{filename}=JAN12_CHINAFISH_LZ_003.RAS
{filename_root}=JAN12_CHINAFISH_LZ
{version_name}=Raster Scan V.0.1
{version_number}=1
{pid}=0
{x_motor}=HORZ
{y_motor}=VERT
{region_filename}=TEST.RGN
{start_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:34:08
{stop_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:43:07
{channel_count}=16
{point_count}=240
{raster_count}=301
{count_time}=0.004690
{sweep_count}=1
{ascii_output}=1
{pixels_per_point}=1
{scan_direction}=2
{scan_type}=1
{current_x_direction}=-1
{run_number}=3

{channel:%02d}=00
{channel_name}=
{channel_enabled}=True

{channel:%02d}=01
{channel_name}=
{channel_enabled}=True
...
```

The text surrounded by curly-braces represent substitution templates which can be used in rasextract's *-t* and *-o* options. For example, to create TIFF output files consisting of the scan date and channel number formatted as a two-digit decimal with leading zeros one could use the following command line:

```
$ rasextract -o "{start_time:%Y-%m-%d}_{channel:02d}.tiff" JAN12_CHINAFISH_LZ_003.RAS channels.txt
Writing channel 1 (Al) to 2012-01-17_01.tiff
Writing channel 2 (Si) to 2012-01-17_02.tiff
Writing channel 3 (P) to 2012-01-17_03.tiff
Writing channel 4 (S) to 2012-01-17_04.tiff
Writing channel 5 (Cl) to 2012-01-17_05.tiff
Writing channel 6 (ES) to 2012-01-17_06.tiff
Writing channel 7 (Ca) to 2012-01-17_07.tiff
Writing channel 9 (HHH) to 2012-01-17_09.tiff
Writing channel 10 (Cr) to 2012-01-17_10.tiff
```

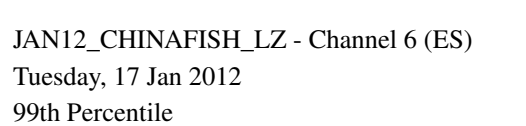
In addition to the templates available from the RAS header, other templates are available which are derived from the rasextract command line. These are named after the command line parameter they represent and include:

- `{percentile}` - The percentile limit applied to the data
- `{interpolation}` - The interpolation algorithm used when rescaling the image
- `{crop}` - The crop coordinates specified
- `{colormap}` - The colormap selected for the image
- `{output}` - The output filename for the image (only available for use with `--title`)

Quite complex titles can be achieved with this syntax. For example:

```
{filename_root} - Channel {channel} ({channel_name})\n{start_time:%A, %d %b %Y}\n{percentile:g}th
```

Will produce titles like this within the image:



Note that the backslash-n (\n) escape sequence was used to generate line-breaks within the template.

Advanced Usage

When combined with some simplistic bash scripting (under Linux) quite complex sequences can be achieved. For example, if one wished to extract a set of channels from a RAS file into TIFF files, rendering each at a range of different percentiles, with axes and a title reflecting the channel and the percentile, one could use the following command line:

```
$ for pct in 100 99.9 99 95 90
> do rasextract -p $pct -a -o "fish_C{channel:02d}_P{percentile}.tiff" -t "Channel {channel} - {c
> done
Writing channel 1 (Al) to fish_C01_P100.0.tiff
Writing channel 2 (Si) to fish_C02_P100.0.tiff
Writing channel 3 (P) to fish_C03_P100.0.tiff
Writing channel 4 (S) to fish_C04_P100.0.tiff
Writing channel 5 (Cl) to fish_C05_P100.0.tiff
Writing channel 6 (ES) to fish_C06_P100.0.tiff
Writing channel 7 (Ca) to fish_C07_P100.0.tiff
Writing channel 9 (HHH) to fish_C09_P100.0.tiff
Writing channel 10 (Cr) to fish_C10_P100.0.tiff
Writing channel 1 (Al) to fish_C01_P99.9.tiff
Writing channel 2 (Si) to fish_C02_P99.9.tiff
Writing channel 3 (P) to fish_C03_P99.9.tiff
Writing channel 4 (S) to fish_C04_P99.9.tiff
Writing channel 5 (Cl) to fish_C05_P99.9.tiff
Writing channel 6 (ES) to fish_C06_P99.9.tiff
Writing channel 7 (Ca) to fish_C07_P99.9.tiff
Writing channel 9 (HHH) to fish_C09_P99.9.tiff
Writing channel 10 (Cr) to fish_C10_P99.9.tiff
Writing channel 1 (Al) to fish_C01_P99.0.tiff
Writing channel 2 (Si) to fish_C02_P99.0.tiff
Writing channel 3 (P) to fish_C03_P99.0.tiff
Writing channel 4 (S) to fish_C04_P99.0.tiff
Writing channel 5 (Cl) to fish_C05_P99.0.tiff
Writing channel 6 (ES) to fish_C06_P99.0.tiff
Writing channel 7 (Ca) to fish_C07_P99.0.tiff
Writing channel 9 (HHH) to fish_C09_P99.0.tiff
```

```
Writing channel 10 (Cr) to fish_C10_P99.0.tiff
Writing channel 1 (Al) to fish_C01_P95.0.tiff
Writing channel 2 (Si) to fish_C02_P95.0.tiff
Writing channel 3 (P) to fish_C03_P95.0.tiff
Writing channel 4 (S) to fish_C04_P95.0.tiff
Writing channel 5 (Cl) to fish_C05_P95.0.tiff
Writing channel 6 (ES) to fish_C06_P95.0.tiff
Writing channel 7 (Ca) to fish_C07_P95.0.tiff
Writing channel 9 (HHH) to fish_C09_P95.0.tiff
Writing channel 10 (Cr) to fish_C10_P95.0.tiff
Writing channel 1 (Al) to fish_C01_P90.0.tiff
Writing channel 2 (Si) to fish_C02_P90.0.tiff
Writing channel 3 (P) to fish_C03_P90.0.tiff
Writing channel 4 (S) to fish_C04_P90.0.tiff
Writing channel 5 (Cl) to fish_C05_P90.0.tiff
Writing channel 6 (ES) to fish_C06_P90.0.tiff
Writing channel 7 (Ca) to fish_C07_P90.0.tiff
Writing channel 9 (HHH) to fish_C09_P90.0.tiff
Writing channel 10 (Cr) to fish_C10_P90.0.tiff
```

1.4 rasinfo

This utility accepts a source RAS file from QSCAN. It extracts and prints the information from the RAS file's header. If the optional channels definition file is also specified, then channels will be named in the output as they would be with rasextract.

1.4.1 Synopsis

```
$ rasinfo [options] data-file [channels-file]
```

1.4.2 Description

Output information from the header of *data-file*. The optional *channel-file* defines the indices and names of the channels. If the *channel-file* is omitted channels in .RAS files will be unnamed.

--version
show program's version number and exit

-h, --help
show a help message and exit

-q, --quiet
produce less console output

-v, --verbose
produce more console output

-l LOGFILE, **--log-file**=LOGFILE
log messages to the specified file

-P, --pdb
run under PDB (debug mode)

-e, --empty
if specified, include empty channels in the output (by default empty channels are ignored)

-t, --templates
output substitution templates use with *rasextract --title* and *rasextract --output*

-c, --channels
output information about individual channels in addition to header details (note: this requires reading the entire file which can take some time)

1.4.3 Examples

Basic Usage

The following is an example of basic usage of rasinfo, including `-r` switch to output channel count ranges:

```
$ rasinfo -r JAN12_AMNHBIRD_HZ_004.RAS
File name:              JAN12_AMNHBIRD_HZ_004.RAS
Original filename:      JAN12_AMNHBIRD_HZ_004.RAS
Original filename root: JAN12_AMNHBIRD_HZ
Version name:           Raster Scan V.0.1
Version number:         1
PID:                    0
X-Motor name:           HORZ
Y-Motor name:           VERT
Region filename:        TEST.RGN
Start time:             Tuesday, 17 January 2012, 07:06:05
Stop time:              Tuesday, 17 January 2012, 13:00:33
Channel count:          16
Channel resolution:     3400 x 1301
Count time:             0.003987
Sweep count:            1
Produce ASCII output:   1 (Yes)
Pixels per point:       1
Scan direction:         2 (+ve and -ve)
Scan type:              1 (Quick scan)
Current X-direction:    -1
Run number:             4
Channel  0 range:       0-0 (empty)
Channel  1 range:       0-2449
Channel  2 range:       0-1159
Channel  3 range:       0-907
Channel  4 range:       0-944
Channel  5 range:       0-900
Channel  6 range:       0-1507
Channel  7 range:       0-328
Channel  8 range:       0-349
Channel  9 range:       0-432
Channel 10 range:       0-359
Channel 11 range:       0-394
Channel 12 range:       0-270
Channel 13 range:       0-3989
Channel 14 range:       0-222
Channel 15 range:       0-1372

Comments:
The comment line always goes in speech marks
like this
and this
line 4
line 5
and the final line
```

Substitution Templates

The *-t* option causes rasinfo to output the same data but in a form suitable for use as substitution templates in *rasextract -t* and *rasextract -o* options:

```
$ rasinfo --templates JAN12_CHINAFISH_LZ_003.RAS
{rasfile}=JAN12_CHINAFISH_LZ_003.RAS
{filename}=JAN12_CHINAFISH_LZ_003.RAS
{filename_root}=JAN12_CHINAFISH_LZ
{version_name}=Raster Scan V.0.1
{version_number}=1
{pid}=0
{x_motor}=HORZ
{y_motor}=VERT
{region_filename}=TEST.RGN
{start_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:34:08
{stop_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:43:07
{channel_count}=16
{point_count}=240(sandbox)dave@morpheus:~/Desktop/Beamline/Beamline 6-2/data/data sorted by sample
{rasfile}=JAN12_CHINAFISH_LZ_003.RAS
{filename}=JAN12_CHINAFISH_LZ_003.RAS
{filename_root}=JAN12_CHINAFISH_LZ
{version_name}=Raster Scan V.0.1
{version_number}=1
{pid}=0
{x_motor}=HORZ
{y_motor}=VERT
{region_filename}=TEST.RGN
{start_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:34:08
{stop_time:%Y-%m-%d %H:%M:%S}=2012-01-17 21:43:07
{channel_count}=16
{point_count}=240
{raster_count}=301
{count_time}=0.004690
{sweep_count}=1
{ascii_output}=1
{pixels_per_point}=1
{scan_direction}=2
{scan_type}=1
{current_x_direction}=-1
{run_number}=3

{channel:%02d}=00
{channel_name}=
{channel_enabled}=True

{channel:%02d}=01
{channel_name}=
{channel_enabled}=True

{channel:%02d}=02
{channel_name}=
{channel_enabled}=True


{raster_count}=301
{count_time}=0.004690
{sweep_count}=1
{ascii_output}=1
{pixels_per_point}=1
{scan_direction}=2
{scan_type}=1
{current_x_direction}=-1
```

```
{run_number}=3

{channel:%02d}=00
{channel_name}=
{channel_enabled}=True

{channel:%02d}=01
{channel_name}=
{channel_enabled}=True

{channel:%02d}=02
{channel_name}=
{channel_enabled}=True
...
```

1.5 rasviewer

This is the GUI portion of the rastools suite. It can open multiple data files simultaneously, displaying one channel of data from each at a time, export images of those channels and perform all the manipulations that **rasextract** is capable of.

rasviewer has no (special) command line options beyond those of any ordinary X11 application and is usually invoked from the start menu of whatever platform it is installed upon.

1.5.1 Opening Files

XXX To be written

1.5.2 Manipulating Channels

XXX To be written

1.5.3 Exporting Images

XXX To be written

1.6 License

CHAPTER 2

Indices and tables

- genindex
- search

Symbols

R